NUCLEAR SCIENCE AND TECHNIQUES **25**, 010602 (2014)

A neural network to predict reactor core behaviors*

Juan José Ortiz-Servin,[1,†] David A. Pelta,[2] and José Alejandro Castillo[1]

[1]*Instituto Nacional de Investigaciones Nucleares, Carretera Mexico Toluca S/N, La Marquesa Ocoyoacac, Estado de Mexico, CP 52750, Mexico*

[2]*ETS Ingeniería Informática y Telecomunicaciones, Universidad de Granada, C/Daniel Saucedo Aranda, s/n 18071, Granada, Spain*

(Received October 15, 2013; accepted in revised form December 6, 2013; published online February 20, 2014)

The global fuel management problem in BWRs (Boiling Water Reactors) can be understood as a very complex optimization problem, where the variables represent design decisions and the quality assessment of each solution is done through a complex and computational expensive simulation. This last aspect is the major impediment to perform an extensive exploration of the design space, mainly due to the time lost evaluating non promising solutions. In this work, we show how we can train a Multi-Layer Perceptron (MLP) to predict the reactor behavior for a given configuration. The trained MLP is able to evaluate the configurations immediately, thus allowing performing an exhaustive evaluation of the possible configurations derived from a stock of fuel lattices, fuel reload patterns and control rods patterns. For our particular problem, the number of configurations is approximately 7.7×10^{10}; the evaluation with the core simulator would need above 200 years, while only 100 hours were required with our approach to discern between bad and good configurations. The later were then evaluated by the simulator and we confirm the MLP usefulness. The good core configurations reached the energy requirements, satisfied the safety parameter constrains and they could reduce uranium enrichment costs.

Keywords: Boiling Water Reactors (BWRs), Neural Networks, Optimization

DOI: 10.13538/j.1001-8042/nst.25.010602

I. INTRODUCTION

The global fuel management problem in BWRs (Boiling Water Reactors) can be understood as a very complex optimization problem, where the variables represent design decisions and the quality assessment of each solution is done through a complex and computational expensive simulation. This last aspect is the major impediment to perform an extensive exploration of the design space, mainly due to the time lost evaluating non promising solutions.

In a previous work [1], we presented a Recurrent Neural Network (RNN) to find good configurations from several stocks of optimized solutions to fuel lattice design, fuel load pattern design and control rod patterns design. These partial solutions to the global fuel management problem are combined to find a core configuration of fresh fuel bundles, a fuel reload pattern and core exposition calculus are made through control rod patterns in several burnup steps in the cycle length. SIMULATE-3 [2] core simulator was used to calculate the reactor behavior of those configurations. So, thermal limits, throughout of the cycle and cold Shutdown Margin (SDM) at the beginning of the cycle are calculated in order to determine the quality of the configurations. Fuel lattices with several average uranium enrichments were used in the fuel lattice stock. In the first instance (Ref. [1]), it was possible to find good configurations with average uranium enrichments lower than a reference case.

Fuel lattices stock was created by Neural Networks (NN) [3] and Path Relinking [4] techniques. Both optimization techniques use CASMO4 [5] to calculate the lattice parameters: local power peaking factor and a reactivity value, both at the beginning of the fuel lattice life. Fuel reloads were generated using NN [6] and Tabu Search [7]. Both optimization techniques use SIMULATE-3 to calculate the end of cycle under Haling condition [8]. Finally, control rod patterns were generated by Tabu Search [9] and Ant Colony System [10] optimization techniques. SIMULATE-3 was used to determine thermal limits throughout the cycle.

In this contribution our aim is to address the following research questions:

1) Is it possible to design a surrogate model of the simulator that allows performing a fast discrimination between good and bad configurations?

2) Having the previous model, would it be possible to evaluate the set of all the potential configurations arising from the combinations of alternatives in the stocks available?

In order to address these questions, we propose a Multi-Layer Perceptron (MLP) as a simplified model of the simulator to discern between bad or good core configurations. In Ref. [11], an adaptive classifier model is used to solve optimization problems. The classifier eliminates non-feasible solutions reducing the cpu time to solve the problem. In our paper, the MLP eliminates bad core configurations.

The rest of the paper is organized as follows. In Sec. II we briefly describe the MLP concepts. Then in Sec. III, we show how the MLP was trained. In Sec. IV and Sec. V, we show some practical results with the trained MLP. Finally, conclusions and references are shown.

* Supported in part by Campus CEI-BioTic GENIL, from University of Granada. D. Pelta acknowledges support from Projects TIN2011- 27696-C02-01 from the Spanish Ministry of Economy and Competitiveness and P11-TIC-8001 from Andalusian Government. The authors gratefully acknowledge the Departamento de Gestión de Combustible of the Comisión Federal de Electricidad of México, the support given by CONACyT from Mexico, through the research project CB-2011-01-168722 and the ININ through the research project CA-215

† Corresponding author, juanjose.ortiz@inin.gob.mx

II. MULTI-LAYER PERCEPTRON NEURAL NETWORK

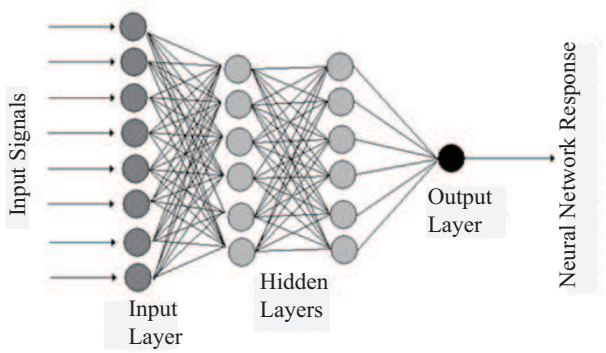

Fig. 1. Typical MLP Architecture.

The artificial neural network is a computer model. It can be used to pattern recognize, prediction, memory, etc. There are several neural network models. One of the most popular is the Multi-Layer Perceptron (MLP) [12]. In Fig. 1, we show the typical architecture of this kind of neural network.

The neural network is composed by one input layer, one or more hidden layers and one output layer. Input layer collects external information and distributes it to hidden layers. Hidden layers process the information and output layer shows results. Each layer has several neurons. Neuron is the lowest information processor. The neuron makes a weighted sum of all its input signals:

$$I_i = \sum w_{ij} x_j, \tag{1}$$

where I_i is the net input to i-th neuron, w_{ij} is the weight connection between i-th neuron (in a previous layer) and j-th neuron (in a current layer), x_j is the signal between both neurons. Then net input is converted to an activation signal according to the activation function $f(I_i)$:

$$A_i = fI_i. \tag{2}$$

Activation function gives a trigger threshold for the neuron. If net input is lower than the threshold, the neuron is inhibited. If net input is greater than the threshold, the neuron is excited. These inhibitory or excitatory signals are propagated by all neurons in the network until a global response is generated.

Weights connection between neurons must be adjusted in order that MLP response becomes adequate to the input signal. This process is named neural network training. Back propagation is the most popular training algorithm used for MLP. First, the input signal is passed through of layers until a response is generated. Second, the response is compared with the desired output and an error signal is generated. Third, the error signal is back propagated to first hidden layer, updating weight connections. The process is repeated until the error signal is lower than the tolerance.

III. DATA SETS AND TRAINING PROCESS

Neural network training was made using a set of 2680 samples. An explanation about how these samples were obtained will be shown in the next section. Each sample is a pair of input and output vectors where the former is a possible core reactor configuration (partial solutions to the global problem), and the latter is a set of core safety parameters (thermal limits, k_{eff} and SDM) which are calculated by SIMULATE-3 for that core reactor configuration. The values in the output vectors are aggregated into a single real value.

Input vectors:

For this study an 18-month equilibrium fuel cycle is used. The fuel reload has two fresh fuel batches. Both fresh fuel batches have a similar axial design: one node of natural uranium at the bottom, 8 nodes with 4.01% U235 and variable gadolinia concentration, 6 nodes with 4.01% U235 and high gadolinia concentration, 8 nodes with 3.96% U235 and high gadolinia concentration. Finally, two nodes of natural uranium at the top of fuel bundle.

A variable number of fuel lattices for three segments of both fresh fuel batches were generated by Path Relinking and Neural Networks. Fuel reloads were generated using Neural Networks and Tabu Search. Control rod patterns were generated by Tabu Search and Ant Colony System optimization techniques.

Input vectors are represented by an 8-entry array. First six entries are used to represent 3 axial segments of both fresh fuel batches. Entry number seven is used to specify a fuel loading pattern. Finally, the last entry is used to specify a set of control rod patterns throughout of the cycle. For all entries, integer numbers are used to specify a fuel lattice or a fuel reload or control rod patters according to the list. An example of input vector is shown in Fig. 2.

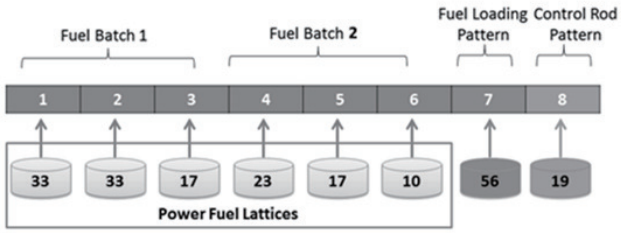

Fig. 2. An example of input vector or core reactor configuration. The number of alternatives available for each entry is also shown. For example, 56 alternatives are available for entry 7.

Each input vector is unique and defines a particular core reactor behavior. According to the size of lists used in this work, the universe of possible solutions to this problem is:

$33\times33\times17\times23\times17\times10\times56\times19 \approx 7.7\times10^{10}$.

Output vector:

In order to construct the output vector we proceed as follows. An input vector is introduced into SIMULATE-3 in order to do several runs and to obtain thermal limits (MFLCPR,

MFLPD and MAPRAT), k_{eff} throughout of the cycle and cold shutdown margin at the beginning of the cycle. These core parameters are satisfied if they fulfill the following constraints:

1. limiting fractions to Linear Heat Generation Rate (MFLPD) <0.93
2. limiting fractions to Critical Power Ratio (MFLCPR) < 0.93
3. limiting fraction to Average Planar Linear Heat Generation Rate (MAPRAT) < 0.93
4. $k_{\text{eff}}-$ target k_{eff} <400 pcm (1 pcm $= 10^{-5}$)
5. cold shutdown margin >0.01.

Some reactor core configurations may fulfill all or some of the safety parameters. For the purposes of this work, if is not fulfilled in only one burnup step, then it is considered like not globally fulfilled. The same is applied for thermal limits. Then, the number of safety parameters fulfilled can be determined for each core reactor configuration (input vector). The value in the output vector is calculated as a function of the number of core parameters fulfilled (CPF):

$$\text{Output Value} = e^{-(1-\frac{\text{CPF}}{5})}. \tag{3}$$

When CPF is equal to zero, Output Value is e^{-1}. When CPF is 5, Output Value is $e^{0} = 1$.

The MLP will predict the Output Value, and then, we can use such prediction to calculate the number of core parameters fulfilled.

The neural network has three layers: an input layer with 8 neurons, a hidden layer with 4 neurons and the output layer with only one neuron. The number of neurons in hidden layers was determined by analyzing the neural network behavior for sizes: 3, 4, 5 and 6 neurons. The best results were obtained for 4 neurons, so that is the value kept for the rest of the paper.

IV. EXPERIMENTS AND RESULTS

The 2680 samples in the dataset were divided in two subsets: training set and test set. Training set has 70% of all samples and test set has the rest 30%. Both subsets were randomly created from the original one. MLP was trained with the back-propagation algorithm using the Generalized Delta Rule for weights updating. The program BackProp [13] was used to train the MLP. Figs. 3 and 4 show the CPF values for training and test sets, respectively. As the MLP predicts the Output Value, the corresponding CPF is calculated using the inverse function of Eq. (3). Please note that CPF values calculated from MLP predictions are continuous values, while CPF values calculated from SIMULATE-3 are discrete values. In case of perfect learning, a 45° line should be observed in both figures. The results show that MLP is able to roughly distinguish between very bad or very good configurations (those having low or high CPF values). In other words, MLP almost never classified a very good configuration (according to SIMULATE-3) like a very bad configuration.

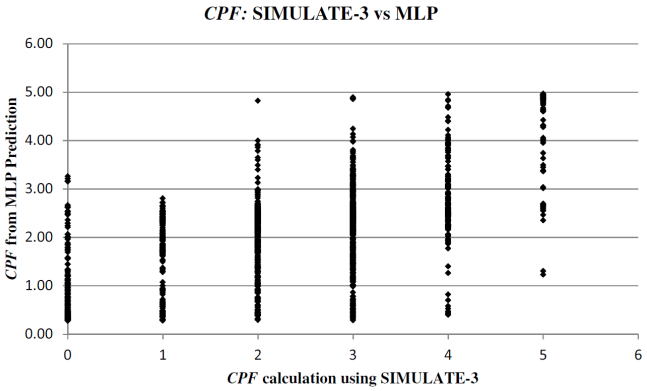

Fig. 3. MLP results for training set.

To further analyze the results, we will consider the following questions: how many core configurations are recognized as good configurations by MLP but SIMULATE-3 says they are bad configurations? And, in turn, how many good configurations (according to SIMULATE-3) would be discarded by MLP?

If we take a threshold value like 4.5 in the MLP's predicted Output Value, we can consider as "good" configurations those that are above the threshold and as "bad" those that are below. Truly good configurations are those with CPF $= 5$ according to SIMULATE-3.

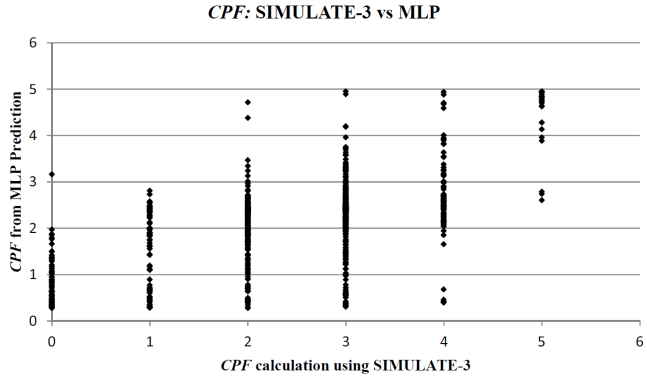

Fig. 4. MLP results for test set.

Now, taking the problem as a binary classification one, Table 1 shows the so called Confusion Matrix for training and test sets. This matrix indicates the amount of True Positives (TP, good core configurations according to SIMULATE-3 and MLP), True Negatives (TN, bad core configurations according to SIMULATE-3 and MLP), False Negatives (FN, good core configurations according to SIMULATE-3, but MLP classifies them as bad ones) and False Positives (FP, bad core configurations according to SIMULATE-3, but MLP classifies them like good ones) obtained.

MLP learnt to classify core configurations with acceptable confidence. As shown in Table 1, the number of False Negatives is high and it means that MLP could discard an important number of good core configurations. On the other hand, a low

TABLE 1. Confusion matrix for good core configurations

Cycle	Training Set		Test Set	
	Good Core Configurations according to SIMULATE-3	Bad Core Configurations according to SIMULATE-3	Good Core Configurations according to SIMULATE-3	Bad Core Configurations according to SIMULATE-3
Good Core Configurations according to MLP	53 (TP)	12 (FP)	17	9
Bad Core Configurations according to MLPP	32 (FN)	1762 (TN)	8	787 (TN)

number of False Positive means that a few bad core configurations are considered as good core configurations.

Two additional statistical measures of the performance of a binary classification test can be calculated from the confusion matrix, namely sensitivity and specificity. Their definitions are:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{4}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

Sensitivity (also called recall rate in some fields) measures the proportion of actual positives which are correctly identified as such. Specificity measures the proportion of negatives which are correctly identified. A perfect predictor would be described as having 100% sensitivity and 100% specificity. In our case, the results are as follows:

TABLE 2. Sensitivity and Specificity of the binary classification based on the NN

	Sensitivity (%)	Specificity (%)
Training set	62.35	99.32
Test set	68.00	98.88

In other words, the MLP is excellent for distinguishing bad configurations but not so good at detecting the good ones. However, we should recall that our aim is to explore the whole universe of solutions and, in order to do this we should avoid the full evaluation (with SIMULATE-3) of bad or not promising configurations. This process is described in the next section.

V. MLP USED TO FIND GOOD CORE CONFIGURATIONS

The trained MLP was used as a filter in the process of exhaustive enumeration of possible configurations. Given a configuration, we evaluate it with the MLP and if the predicted output value is greater than certain threshold, then the configuration is considered as potentially good and is archived for a later evaluation with SIMULATE-3.

Table 3 shows, for a given threshold value, the number of core configurations that were considered potentially good, how many of them were effectively good (according to SIMULATE-3), and the relation between both values.

Using a threshold value of 4.5 gave around 580 000 core configurations, being less than 1% of them, effectively good. Using a higher threshold effectively reduced the configurations that passed the filter and increased the percentage of effectively good configurations.

TABLE 3. Results of the exhaustive enumeration process. The total number of core configurations was 7.7×10^{10}

Threshold Value	Core configurations with Output Value greater than the threshold (A)	Core configuration with CPF = 5 according to SIMULATE-3 (B)	(B/A)×100 %
4.95	587570	1141	0.19
4.97	33279	366	1.10
4.98	930	48	5.16

VI. CORE CONFIGURATIONS ANALYSIS

In this section we will analyze the configurations obtained after the enumeration process. The study was made for an equilibrium BWR cycle of 18 months, with a cycle length of 10 896 MWD/T at full power conditions. For this cycle exposure the target k_{eff} is set to 0.9978.

The fuel reload has two fresh fuel bundle batches. The first one (Batch A) has an average uranium enrichment of 3.66%, 10 gadolinia rods and a batch size of 60 fuel bundles. The second batch (Batch B) has the same average uranium enrichment and 8 gadolinia rods and a batch size of 52 fuel bundles. An uranium requirement (UR) for this fuel reload can be defined in the following way:

$$\text{UR} = 60 \times U_{\text{FB-A}} + 52 \times U_{\text{FB-B}} \tag{6}$$

$U_{\text{FB-A}}$ is the average uranium enrichment for fuel Batch A;
$U_{\text{FB-B}}$ is the average uranium enrichment for fuel Batch B

The baseline for the comparison is a reference core configuration with UR= 409.92% (applying Eq. (6) and $k_{\text{eff}-\text{EOC}} = 0.9978$. Core configurations with lower UR value means they save uranium with respect to the reference one.

From the results shown in Table 3, we have available 1141 core configurations with CPF = 5. From this set, we will consider only 165 configurations having a $k_{\text{eff}-\text{EOC}}$ greater than the reference one.

Figure 5 shows a scatter plot where each point represents a core configuration. The X axis is the uranium saving (according to Eq. (6) and the reference UR = 409.92%) , while the Y axis indicates the difference against the k_{eff} reference value.

TABLE 4. Hamming distances between the reference configuration and the selected ones

Entry	1	2	3	4	5	6	7	8
Configs	22	158	3	5	58	0	57	0
% (over 165)	13.33	95.76	1.82	3.03	35.15	0.00	34.55	0.00

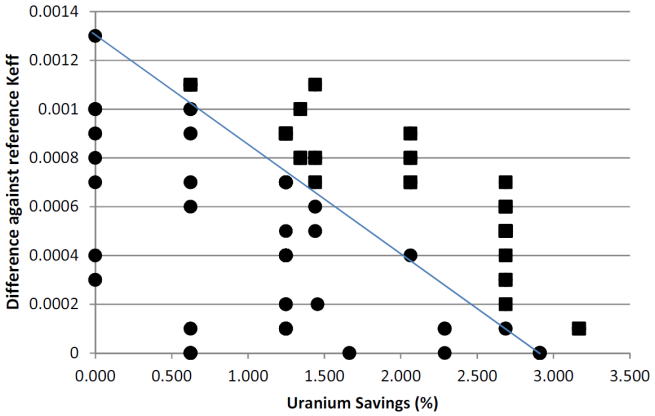

Fig. 5. (Color online) Core configurations performance. Solid circles means good core configurations, solid squares means core configuration with economical advantages with respect to the reference one.

It is clear from the plot that there are better configurations than the reference one, both improving k_{eff} and uranium savings. Core configurations above the solid line and marked with solid squares are those that decrease the uranium enrichment of one of the fresh fuel batches without loss of energy production. These core configurations could have economical advantages with respect to the reference one, both by uranium savings and electrical energy sales. Core configurations under the line (in solid circles) are good core configurations without economical advantages with respect to the reference one.

The core configurations analysis can also be done from other points of view:

In first place, we measured the Hamming distance [14] between the reference core configuration and the selected ones. We obtained 35 configurations with one position changed; 122 with two positions changed and 8 configurations with three changes.

Then, in order to analyze where those changes happened, i.e. what are the necessary changes in the reference configuration to obtain a better solution, we counted for every entry the number of configurations with a different value than that of the reference configuration (we must remember that the reference configurations is [1,1,1,1,1,1,1,1]). The maximum value for an entry is 165 stating that all the configurations under analysis have a different value than the reference one. The results are shown in Table 4.

It is clear that the entries with greater variability are 2, 5 and 7; there were no configurations with other alternatives than the reference one for entries 6 and 8. In Table 4, it is clear that the reference fuel lattice number two of both fuel batches can be improved. Almost all good core configurations have different fuel lattices in both batches. Also, both fuel lattices are responsible of the uranium savings. More energy production is due to use of another fuel reload against of the reference one.

VII. CONCLUSION

From this work several comments can be made:

- It was possible to train a MLP able to catalog good core configurations. A core configuration is a combination of fuel lattices, control rod patterns throughout of the cycle and a fuel reload.
- The MLP training required around one hour to learn the desired behavior in an AMD processor at 2.1 GHz and a RAM of 1.5 Gb. This small time interval converts the trained MLP as an excellent tool to be coupled with an optimization system to find the best core configuration. The trained MLP could be used like a "bad configuration filter" to avoid running a time expensive 3D core simulator.
- The utility and the quality of the trained MLP was demonstrate by an analysis of results in both test and training sets and their performance when it was coupled with an exhaustive searching algorithm.
- A balance between the required time to evaluate core configurations and the analyzed solution space size can be made by filter threshold adjusting. Lower threshold values are able to help the coupled system to explore more solutions in the optimization process. High threshold values reduce the required CPU time to find a good core configuration.
- This coupled system found several good core configurations with any of these advantages (or both): they overcome the energy requirements with the same uranium enrichment, like the reference case or they reach the energy requirements for decreasing the uranium enrichment. Several core configurations satisfy both scenarios.
- Table 4 tells us about the performance of optimization designs. We can say that, we are able to design fuel reloads, trying the problem like an independent one, with an acceptable confidence. Similar ideas can be said for fuel lattices design. On the other hand, control rod patterns designs have poor performances when they are optimized like an independent problem of the rest of the integral optimization one.

[1] Ortiz J J, Castillo J A, Pelta D A. Nucl Eng Des. 2011, **241**: 3729–3735.
[2] Dean D W. SIMULATE-3 Advanced Three-Dimensional Two-Group Reactor Analysis Code, SSP-95/15 - Rev 3, Studsvik Scandpower, 2005.
[3] Ortiz J J, Castillo A, Montes J L, *et al*. Nucl Sci Eng, 2009,**162**: 148–157.
[4] Castillo A, Ortiz J J, Perusquía R, *et al*. Prog Nucl Energ, 2011, **53**: 368–374.
[5] Rhodes J and Edenius M. CASMO-4 A Fuel Assembly Burnup Program, SSP-01/400 Rev 4. Studsvik Scandpower, 2004.
[6] Ortiz J J and Requena I. Ann Nucl Energy, 2004, **31**: 789–803.
[7] Castillo J A, Alonso G, Morales L B, *et al*. Ann Nucl Energy, 2004, **31**: 151–161.
[8] Haling R K. Operational Strategy for Maintaining an Optimum Power Distribution through Core Life. Proc. ANS Topl. Mtg. Nuclear Performance of Core Power Reactors, TID-7672. US Atomic Energy Comission, 1964.
[9] Castillo A, Ortiz J J, Alonso G, *et al*. Ann Nucl Energy, 2005, **32**: 741–754.
[10] Ortiz J J and Requena I. Ann Nucl Energy, 2006, **33**: 30–36.
[11] Tenne Y. Eng App Artifi Intel. 2012, **25**: 1009–1021.
[12] Caudill M and Buttler C. Understanding neural networks: Computer explorations. V. 1 Basic network. MIT Press (USA), 1994.
[13] Tveter D R. User Manual of Student Version Basis of AI Backprop. Copyright (c) 1990–97.
[14] Zwillinger D (Editor). Standard Mathematical Tables and Formulae. Chapman & Hall/CRC New York (USA), 2003.